UNITED STATES PATENT APPLICATION

FOR

SYSTEM AND METHOD FOR APPLICATION-LEVEL SECURITY

INVENTORS:
JAMES DE PERNA
IZABELLA GALINSKY
LENARD GASSMAN

PREPARED BY:
MCDERMOTT, WILL & EMERY
600 13TH STREET, N.W.
WASHINGTON, DC 20005
(202) 756-8600

SYSTEM AND METHOD FOR APPLICATION-LEVEL SECURITY

RELATED APPLICATIONS

This application relates to and claims priority from Provisional U.S. Application Serial No. 60/248,569 filed November 16, 2000 entitled APPLICATION SECURITY DATABASE, the disclosure of which is hereby incorporated in its entirety by reference.

FIELD OF THE INVENTION

The present invention relates to a computer software application and more particularly to an application for protecting software applications and their underlying proprietary data.

BACKGROUND OF THE INVENTION

Security of computing resources is an ever-growing concern, especially in today's distributed computing environment in which user's can attempt to access resources from various geographical locations.

One type of security application software functions by restricting user's abilities to access physical resources. For example, use of certain terminals may be prohibited, use of certain disk drives may be restricted, and the use of certain printers and other output devices can be prevented. These types of systems provide gross levels of selectivity when prohibiting access to resources.

Many modern operating systems now provide security features which try to prevent unauthorized activity at a finer level of selectivity. Within a physical device, for example, a user may be prohibited from executing certain programs, seeing various directory listings, or

overwriting particular files. These types of security benefits have a number of drawbacks. First, the secured entities are defined at the operating system level. In a file, for example, of customer account balances there are some accounts a user has a business need to access and other accounts that should be protected. At the operating system level, however, the entire file is either accessible or not. Secondly, enforcement of operating systems type security features requires that a user actually login to the computing system which is providing the resource being protected.

There is a need, heretofore unmet by conventional security applications, to protect software applications and their underlying proprietary data, particularly in a manner which grants and controls access to application functionality for a multiplicity of different organizations.

SUMMARY OF THE INVENTION

These and other needs are met by aspects of the present invention which relates to software applications having a hierarchy of functions, sub-functions and sub-sub-functions and made available to one or more clients. The ability of the clients to utilize the various functionality of the applications is controlled by an application security database system (ASDS) which maintains a database of application function hierarchies and client entitlements. The applications consult with the ASDS to determine whether a client's user is authorized to perform a requested function and either performs, or fails to perform, the requested function based on the reply from the ASDS. In particular, rules regarding access to proprietary data associated with different functionality are also maintained by the ASDS; proprietary data is data that is sensitive in some manner (e.g., for business-related reasons, duty of confidentiality, etc.) such that unlimited access to the data should be avoided. The client entitlements are, in some

embodiments, associated with the end-users of the clients not by user name but, rather, by user roles reflecting the business structure of the client. In particular, aspects of the ASDS:

a) provides a security solution to organizations whose application end-users span a large and diverse number of external clients;

b) secures software applications that run on mainframes and distributed systems;

c) segregates entitlements and administration rights by client identity;

d) performs authorization checks for an end-user who is not signed on to the operating system (i.e., the end user could be accessing a remote system);

e) grants access and authorization to applications and their functionality based upon an end-user's set of job roles;

f) provides a web-based front end to ease and simplify administration;

g) permits real-time auditing; and

h) performs changes to entitlements and their authorization settings immediately upon the administrator's action, without the need for the end-user(s) to sign-off and sign-on again to effect the change.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is illustrated by way of example, and not by way of limitation, in the figures of the accompanying drawings and in which like reference numerals refer to similar elements and in which:

FIG. 1 illustrates an exemplary environment in which embodiments of the present application security database system operate.

FIG. 2 illustrates an exemplary flowchart depicting the logical flow of an application requesting authorization for performing a particular function according to an embodiment of the present invention.

FIGS. 3 -10 illustrate exemplary interface screens for administering entitlement and authorization data useful by embodiments of the present invention.

FIGS. 12A, 12B and 13 illustrate exemplary interface screens for providing auditing functions according to embodiments of the present invention.

DESCRIPTION OF THE PREFERRED EMBODIMENT

To aid with the understanding of the present invention, exemplary embodiments are presented within the context of a specific environment involving securities trading. In general, however, the invention is applicable to other environments where a software application, or applications, is made available to one or more clients and access to the application's functionality is desired to be controlled.

GENERAL ENVIRONMENT

FIG. 1 shows an exemplary environment in which various aspects of the present invention can operate. In particular, software applications 106 and 108 are available for use by clients 112 and 114. These software applications can include both distributed applications 108 and applications 106 that run on a mainframe. Clients 112 and 114 interface with the applications using conventional and well-known techniques and methods. This interfacing can be accomplished using local and wide-area networks, the Internet, dedicated terminals, dial-up connections, wireless protocols, and other functionally equivalent alternatives.

In operation, the applications 106 and 108 sometimes utilize external data 110 as well as internal hard-coded data. This data 110 can include proprietary data which is locally or remotely stored, data generated on the fly from manipulating pre-stored data, as well as dynamic information such as live data feeds.

In a preferred embodiment, the clients 112 and 114 contract with a provider of the applications 106 and 108 and data 110. The clients 112 and 114 are allowed to access and utilize some or all of the features of some or all of the respective applications 106 and 108 based on particular contractual arrangements with the applications' provider. Typically a client 112 and 114 is an organization that comprises a number of individuals, or end-users. These individuals have different organizational roles and functions within the client's organization and need different aspects of the applications 106 and 108 in order to carry-out their roles. The applications 106 and 108 are, then, software tools that are made available to one or more clients 112 and 114 for use by the client's end-users. One exemplary environment is in the field of securities trading. In such an environment, the applications 106 and 108 are trading tools and applications which different investment service providers (clients 112 and 114) can access. The extent of the data and tools that are accessible by a particular client is determined by a pre-arranged contract.

An Application Security Database System (ASDS) 102 is a software application which communicates with the software applications 106 and 108 to help ensure clients 112 and 114 are allowed access to only those application features and data for which they are authorized. In particular, various functions and sub-functions of the applications 106 and 108 include software routines that query the ASDS 102 before performing that particular function. This type of security aspect is remotely similar, in principle, to the execution of the "ls" command, for

example, in a typical UNIX environment; before displaying the requested directory listing, a portion of the software code which implements the "ls" command checks the USERID and performs the command only within the permissions granted that USERID by the operating system. Similarly, aspects of the present invention relate to custom, or customizable, applications 106 and 108 whose functions include code portions that call the ASDS 102, pass information regarding the user requesting a function, and ensure the user is only allowed to perform authorized functions according to responses returned by the ASDS 102. In particular, different operating system APIs (e.g., MVS/OS 390 APIs) and message-based services (e.g., HTTP, MQ Series, and XML format etc.) can be used for inter-program communications.

The ASDS 102 relies on a database 104 which stores information that correlates different applications and functions with different clients and users. This stored information 104 is consulted by the ASDS 102 when determining whether a requested function can be performed by the requesting user. In a preferred embodiment, the database 104 relates the functions (and sub-functions) of each application 106 and 108 in a hierarchical fashion. This design allows an administrator to suspend access to an application or some of its parts thereby affecting all clients 112 and 114 and groups of clients globally.

The ASDS 102 and its associated database 104 are administered and maintained by one or more administrators that interface with the ASDS 102 through an administration application 116. This application can preferably provide a web-based interface to allow widely distributed administrators to easily effect system maintenance and configuration regardless of physical location. Alternatively this administration application can include a workflow component whereby entitlement requests are made by end-users or their managers and the entitlement changes are automatically applied to the ASDS database 104 once all required electronic

approvals have been obtained. The functions available through the administration application (e.g., adding users, deleting users, defining user permissions, viewing user permissions, defining application functions, etc.) can, themselves, utilize the security features of the ASDS 102 so that different administrators can be permitted to, or prohibited from, performing a limited subset of administrative tasks.

An auditing application 118 is also provided which communicates with the ASDS 102. This auditing application can be used for tracking information regarding attempted unauthorized activity as well as reporting other system information related to the ASDS 102. The database 104, or another auditing-specific database (not shown) can be used to store information utilized by the auditing application 118.

OPERATIONAL LOGICAL FLOW

FIG. 2 illustrates a high-level flow chart of the operation of the ASDS 102. The applications 106 and 108 which utilize the ASDS 102 are assumed to have the capability to track user identity information and be comprised of functions which have security exits that contact the ASDS 102 for determining security authorization.

In step 202, the application 106, for example, receives through an interface some mouse, keyboard, audio or other type of input from the user 112. The input is initiated from a computing device (e.g., workstation, hand-help, laptop, etc.) available to the user 112 which provides output based on the operations of the application 106. In response to this input, the application 106 initiates, in step 204, performance of a function, or sub-function, as indicated by the input.

As part of performing the requested function, the application 106 provides, in step 206, the ASDS 102 with information identifying the requested function and information identifying

the requesting user. The ASDS 102, in step 208, utilizes the information to consult the database 104 in order to determine whether the requesting user is authorized to perform the requested function.

If the requesting user is authorized to perform the requested function, then the ASDS 102 provides, in step 210, a message indicating such authorization to the application 106. If the requesting user is not authorized to perform the requested function, then the ASDS 102 provides, in step 210, a message indicating lack of such authorization to the application 106. In addition, the ASDS 102 logs information about the failed attempt; for example, in database 104. The logged information can include, for example, information about the date and time of the request, the requester's identity, and the requested function's identity.

The above description explicitly identifies only applications and functions within applications. However, a function can include sub-functions which, in-turn, include sub-sub-functions, etc., each of which can have their own individual authorization parameters stored in database 104. The present invention contemplates applications 106 and 108 with multiple levels of functions; however, the following detailed examples happen to describe 4 levels of functions and sub-functions for illustrative purposes only.

Furthermore, many functions provide fields of data for display and editing to a client 112. These fields, from the database 110 for example, can involve their own authorization parameters as well. Therefore, within the database 104, each function defined within an application can be defined as owning one or more fields. The application, in performing step 206, can obtain a list of all fields belonging to a requested application function that the requesting user is authorized to access. This list will instruct the calling application as to how the fields should be displayed -- fully-enabled, write-protected, disabled, invisible, etc.

Based on the information the calling application 106 receives from the ASDS 102, the application 106 performs the requested function accordingly, in step 212. The result of the application 106 performing the requested function is then forwarded to the client 112 so that the client can continue using the application 106.

THE ASDS DETAILS

The details of the ASDS 102 are discussed below based on a convenient separation into components focusing on: administration, auditing, and enforcement.

ADMINISTRATION

Although depicted as a separate functional block in FIG. 1, the administrative application 116 can be a co-residing software portion of the ASDS 102. The administrative functions from application 116 are preferably made available through a GUI or other easy-to-use interface. These functions allow security administrators to list, view or maintain security settings and access entitlements (for which they are authorized).

In specifying and maintaining these security parameters, that are stored in the database 104, administrators define the following data:

Roles

Job roles are the mechanism by which application functionality is correlated to end-users at the clients 112 and 114. The administrative application 116 permits the creation of new roles

and attaches them to various clients 112 and 114. Once a role has been created it can be assigned by an administrator to a particular end-user at a client.

One benefit of the ASDS 102 is the flexibility afforded to administrators. For example, the functionality to create a new role may be reserved for only administrators working for the provider of the applications 106 and 108. However, the functionality to assign end-users to a particular role can be delegated to administrators working for the client associated with that particular role. When performing certain functionality, the administrative application 116 checks with the ASDS 102 to ensure the requesting administrator has the necessary authorization. In this way, the ASDS 102 is, itself, used in its own administration.

"Roles" provide a powerful and easy mechanism for defining entitlements and are included in many of the example environments described herein. However, roles are not the only mechanism within the present invention by which application functionality is correlated with end-users. One alternative expressly contemplated within alternative embodiments of the present invention is to associate entitlements with a particular user ID. These alternatives are not mutually exclusive but can even be used conjunctively to provide the ease of modifications offered by roles-based administration while providing the finer levels of selectivity offered through user IDs.

Application Functionality

Applications 106 and 108 are considered by the ASDS 102 to be hierarchically arranged. For example, an application has functions which have sub-functions, which have sub-sub-functions, etc. These hierarchical constructs are logical in nature in that they are not necessarily

distinct physical pieces of contiguous software code but are anything that the application 106 and 108 defines them to be.

For example, a sub-function can be an entire screen, a single button on a screen, or even a list box that displays data from a file. These functions can also have associated "fields" of data elements for display and updating.

Once application functions are defined and stored in the database 104, they can be enabled or disabled for all clients, for particular clients, or for selected job roles at particular clients. Accordingly, a particular job role at client 112 is not necessarily granted the same entitlements as the same job role at client 114 with respect to any particular application functionality.

Within the database 104, the hierarchical relations of the various functions are maintained. Thus, if the authorization attribute for a function is changed from "enabled" to "read-only", then all child sub-functions (and sub-sub-functions) inherit the new "read-only" setting. Also, if a function is set to expire for a particular client (e.g. client 112) at a particular time, then that function expires for all job-roles of that client 112 that previously had access to the function.

These inheritance features enable administrators to control access at various levels and, therefore, make the administration process faster and more efficient. It is especially valuable during times when access must be permanently or temporarily taken away from different segments of the user population in a rapid fashion.

Throughout this disclosure, the term "function" is used for convenience even though it encompasses more than merely an application function. Within this disclosure, a function is intended to refer to functions, sub-functions, sub-sub-functions, proprietary data, and data fields.

Essentially a function is any "secured resource" that an application controls access to through operation with the ASDS 102.

Proprietary Data

The ASDS 102 permits controlling user access to proprietary and confidential business data. For example, in a securities trading environment, client 112 can have one or more offices, each of which has one or more managers, each of which has one or more accounts (identified buy a unique account number).

These different hierarchical levels are referred to as "business parties". If an administrator entitles a user to a business party representing the "client", for example, then that user would obtain full access to all business parties subordinate to the "client (i.e., all offices, all managers, all accounts). Similarly, an access entitlement at the "manager" level would give full access to all data pertaining to that manager, as well as all data pertaining to the accounts "owned" by that manager. The design of the ASDS 102 allows each client to define its own organizational structure consisting of business party names. In addition, business party access entitlements can be attached to sharable, pre-defined profiles which simplify assigning to users identical data access requirements.

When an application 106 and 108 performs a function that retrieves business data, a number of techniques are possible for enforcing the entitlements permissions. For example, the application 106 and 108 can invoke SQL "joins" between the business data 110 and the permissions database 104 so as to limit data retrieval to only those clients, offices, managers, accounts that the user is capable of accessing. Alternatively, the applications 106 and 108 can include specific routines within particular functionality that queries the ASDS 102 regarding

certain requested data. For example, the application 106 and 108 can query the ASDS 102 questions such as "Can user XXXX access data for account YYYY?"

The present invention explicitly contemplates that there may be special accounts that require special access permission. These accounts can be designated such that access to them is not automatically authorized based on the "business party" system described above.

User Information

Through functionality for setting-up new users, security administrators can create new users, purge existing users, reset passwords, modify user attributes (e.g., name, ID number, client number, etc.).

EXEMPLARY Screens

A series of interface screens are described below which illustrate an exemplary interface that permits an administrator to define and maintain authorization settings in the ASDS 102. The present invention is not limited to the specific screen layout or screen fields depicted herein but contemplates within its scope those variations and alternatives within the skill and understanding of an artisan in this field. From these screens the data that is stored in database 104 is created, modified and maintained to facilitate the functioning of the ASDS 102.

FIG. 3 depicts an exemplary interface screen 300 for administering an application. From this screen an administrator can define the functions and sub-functions associated with an application as well as administer the entitlement settings at the application-level (i.e., these settings define the entitlement framework in which all other settings for this application must work within). In this particular example, text box 302 identifies the application as "Advanced

Trade/Order Mgmt System" while the sub-window 304 displays the hierarchical arrangement of the functions and sub-functions within the application. The portion 306 of the screen 300 permits the entitlements to be set for the function "Admin Request".

FIG. 4 depicts an exemplary interface screen 400 for administering entitlements to an application based on the client. From this screen an administrator can define the particular entitlements a particular client is permitted with respect to a particular application. In text box 402, the client's name "DLJ Investment Services" is shown. Below that box, in sub-window 404, a hierarchically arranged list of applications and functions is graphically arranged to permit an administrator to easily select a function to administer. In text region 406, the hierarchical arrangement of the selected function is depicted in a non-graphical manner and dialog box 408 provides selections for the administrator to set access authorizations for the selected function "Detail".

FIG. 5 depicts an exemplary interface screen 500 that permits the administering of entitlements to an application based on a particular job role. Similar to the previous screens, the application hierarchy is depicted as well as a setting screen that allows the actual settings to be made. However, in this screen 500, the defined entitlements are associated with a particular job role 504 and a particular client 502.

FIG. 6 depicts an exemplary interface screen 600 for administering data fields. From this screen 600, an administrator can define those fields 602 associated with an application (e.g., Paradigm) and function 608 (e.g., Equity -Option Order Entry Retail Equity). The field names are listed in the leftmost column 606 and the particular authorization settings are changeable via an "Authorization" column 604. The interface screen 600 can be reached via any of the previous

interface screens so that the field entitlements can be specified according to application, client or role (or a combination).

FIG. 7 depicts an exemplary interface screen for associating roles with a client. The client 706 has associated therewith various roles, listed in window 702, that can be administered (e.g., added, deleted, edited, etc.) using the controls 708. User information relating to the client's users can also be provided in sub-window 704. FIG. 8 shows a slight variation in which a particular role can be selected from the variety of different roles 802 in order to display 804 the names of all the clients who are associated with that particular role.

FIG. 9 depicts an exemplary interface screen for administering the data access granted to a user. For a particular user 904 of a particular client 902, a table 906 is provided that displays the data access definitions for that user. The table identifies those accounts of a particular manager at a particular office of a particular client for which the user 904 has access rights. As mentioned earlier, the definition and hierarchical arrangement of these "business parties" can be delegated to the administrators at the client level so that each client can define their own hierarchy.

For example, a user who is entitled to the Midwest Region of Client ABC is also authorized to access all data belonging to entities that report into the Midwest Region (e.g., zones, district offices, sales offices, etc.). Each client can establish its hierarchical structures based upon its unique business contexts of reporting. For example, Client ABC could set up two separate hierarchical structures – one for revenue reporting and one for expense reporting. A user can then be entitled to access a level of the hierarchy (e.g., Midwest Region) for the purposes of revenue type data, but would not necessarily have access to expense type data belonging to that level. At a more granular level, the entitlements for accessing proprietary data

can be depend on the application 106 and 108 or even the specific functions within an application. In this manner, a user can be authorized different levels of access to proprietary data among the different applications 106 and 108 as well as among the different functions within a particular application.

When an application 106 and 108 calls the ASDS 102 to inquire about a user's access to proprietary data, the application 106 and 108 needs only provide the "business party" level being checked and does not have to provide all the levels which are parents and children of the level being checked. The ASDS 102 can interface with an organizational chart database or other data indicating the parent-child arrangements (provided by the client) and perform all the processing necessary to determine the parent-child relationships of the hierarchy.

FIG. 10 depicts an exemplary screen 1000 for setting-up user information. Through this screen an administrator can create a new user, delete an existing user, or modify a user's attributes. FIG. 11 depicts the same screen 1000 but after the Tab 1102 is selected so that the user's roles can be administered. From this screen, an administrator can associate one or more roles with a user. As shown, the user of screen 1000 has at least four different roles that are considered by the ASDS 102 when determining entitlements and authorizations to application 106 and 108.

SECURITY AUDITING

Through the use of the auditing application 118, administrators or security auditors can query the database 104 in order to investigate the settings of the ASDS 102 from a number of different perspectives and a history of its performance with respect to the application 106 and 108 and with respect to the clients 112 and 114.

In particular, an auditor utilizes the auditing application 118, to query the database 104 (through the ASDS 102) for the purpose of producing lists of entitlements from the perspective of the application 106 and 108, from the perspective of the client 112 and 114, from the perspective of the different roles, or from the perspective of a user name or user ID. Also, a history function within application 118 can provide a list of security violations data and a list of changes to entitlements. These different lists and queries can be refined by allowing the auditor to restrict the lists according to different fields and, furthermore, these lists can be output to printers or exported to other software applications to aid in the viewing and diagnostics associated with the lists.

FIG. 12A depicts an exemplary query screen 1200 which allows an auditor to generate a security audit list according to client name 1202. The auditor has, in this example, also selected to limit the list to only those user's named "Bob" at the client 1202. FIG. 12B depicts an exemplary audit list 1250 that might be produced based on the query of FIG. 12A. Although all fields are not explicitly depicted on the list 1250, the auditor can view user data associated with those users satisfying the search criteria.

FIG. 13 depicts an exemplary screen 1300 that shows a log of all changes to the entitlements data within the database 104. Useful information such as date/time, administrator's name, user's name, type of change, etc. is provided to the auditor. A similar screen can be provided by the ASDS 102 through the auditing application 118 which lists all the security violations that have been logged.

One benefit from the ASDS design for the auditing application 118 is that real-time data is available for review (preferably via a simple web-based interface). Because the ASDS 102 logs user security violations and administration activities as they occur, this data is available

instantaneously after the violation or activity has been captured by the system. The logged data includes all relevant details about the activity or violation and these details can be used as filters, or search parameters, so that data can be selected according to User ID, Client name, Administrator ID, date and time, type of violation, type of activity, etc. Administrators and auditors using administrative application 116 and auditing application 118, respectively, can benefit from the availability of real-time logs to monitor the state of the ASDS 102.

SECURITY ENFORCEMENT

As described earlier with reference to FIG. 2, application 106 and 108 embed calls to the ASDS 102 in order to ascertain a user's ability to access application functionality, secured fields and proprietary business data. Similarly, the administrative application 116 and the auditing application 118 also embed calls to the ASDS 102 in order to assess an administrator's ability to access certain screens and modify certain settings. In exemplary embodiment, the database 104 of the ASDS 102 includes a number of tables managed by a relational database management system. These tables store the information and data depicted in the previously described interface screens and have appropriate key fields such as function, or client, or user name, or role so that the various tables can be consulted by the ASDS 102 in order to determine a requesting user's entitlements.

ADDITIONAL FEATURES

Security By Roles

Through the use of ASDS 102, a manager at client 112 and 114 can create a job role that explicitly defines the scope of responsibilities for client employees. Once this role is defined, an administrator can attach one or more application functions to the role. Employees of the client 112 and 114 who are assigned this job role will automatically be granted access to the applications and functionality defined under this role. Thus, the security determinations and decisions do not need to be repeated for every new employee that arrives, or every employee job change, at a client 112 and 114. An unlimited number of users can be assigned to one role, and an unlimited number of applications and functions can be entitled to one role. Since a role is associated with a specific client, it can be controlled independently without affecting other clients.

Real Time Entitlement Changes

Due to the relational database architecture of ASDS 102, administrative changes to entitlements and their authorization settings take effect immediately upon the administrator's actions, without the need for the end-user to sign-off and sign back on the system.

Entitlement Listings

Some applications 106 and 108 may find it beneficial if the ASDS 102 can, instead of providing a simple "yes" or "no" authorization answer, provides a listing of all the functions, sub-functions, fields, proprietary data, etc. that a user is authorized to access. For example, the application 106 and 108 can have the capability of building dynamic toolbars or cascading

menus. Rather than requiring the application 106 and 108 to query the ASDS 102 regarding each possible function, the application 106 and 108 can query for a "listing". In return, the application 106 and 108 is provided a list of the user's authorized entitlements within the calling application. Of particular benefit is that this listing of entitlements does not necessarily include the entirety of the entitlements granted to the user which are stored in database 104. Because of the manner in which the ASDS 102 associates entitlements among, users, roles, functions, applications, etc. within the relational database 104, the calling application 106 and 108 can include filtering parameters (e.g., filtered according to combinations of specific roles, specific applications, specific access levels, specific functions, etc.) within the query for a listing so that the entitlements returned to the application in the listing are only those entitlements matching the filtering parameters.

Authorization Simulator

Many conventional security products require an application user to be logged into the system hosting the security product. Troubleshooting authorization errors (e.g., at a help desk) is made essentially impossible because the help-desk personnel is logged in under their own user ID and cannot re-create the security environment of the end user. The design of the ASDS 102 permits simplified troubleshooting for an administrator or auditor. A custom troubleshooting application permits the administrator to input a specific user Id and a specific application, function or proprietary data name and then simulates a call from the problematic application using the inputted parameters. The ASDS 102 recognizes the call as a simulation and provides an authorization message in return that allows simple diagnosis of problems reported by end-users. Violations that occur during a simulation are not logged in the database 104.

While this invention has been described in connection with what is presently considered to be the most practical and preferred embodiment, it is to be understood that the invention is not limited to the disclosed embodiment, but on the contrary, is intended to cover various modifications and equivalent arrangements included within the spirit and scope of the appended claims. The invention is capable of other and different embodiments and its several details are capable of modifications in various obvious respects, all without departing from the invention. Accordingly, the drawings and description are to be regarded as illustrative in nature, and not as restrictive.